CHAPTER 3
Bootstrap Layout Components

In addition to all of the markup provided in the previous chapter, Bootstrap provides a toolkit of flexible components that can be used in designing application interfaces, web features, and more. All of the plugins are available in one separate JavaScript file, or you can use the Bootstrap customizer to pick and choose which plugins you want. Personally, on the projects that I build, I lump them all together. That way I have options.

Dropdown Menus

Dropdown menus are toggleable, contextual menus for displaying links in a list format. The dropdowns can be used on a variety of different elements, navs, buttons, and more. You can have a single dropdown or extend the dropdown into another submenu. You can see a basic dropdown menu in Figure 3-1.

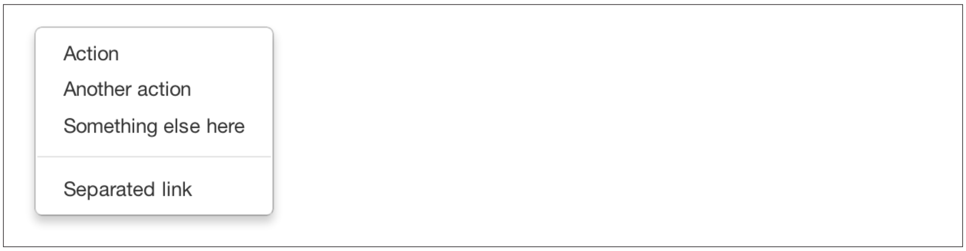


Figure 3-1. Basic dropdown menu

The following code creates a basic dropdown menu:

```
<ul class="dropdown-menu" role="menu" aria-labelledby="dropdownMenu">
  <li><a tabindex="-1" href="#">Action</a></li>
  <li><a tabindex="-1" href="#">Another action</a></li>
  <li><a tabindex="-1" href="#">Something else here</a></li>
  <li class="divider"></li>
```

```
    <li><a tabindex="-1" href="#">Separated link</a></li>
  </ul>
```

Options

Right-align

Add `.pull-right` to a `.dropdown-menu` to right-align the dropdown menu to the parent object:

```
  <ul class="dropdown-menu pull-right" role="menu" aria-labelledby="dLabel">
    ...
  </ul>
```

Submenu

If you would like to add a second layer of dropdowns (see Figure 3-2), simply add `.dropdown-submenu` to any `<li>` in an existing dropdown menu for automatic styling:

```
  <ul class="dropdown-menu" role="menu" aria-labelledby="dLabel">
    ...
    <li class="dropdown-submenu">
      <a tabindex="-1" href="#">More options</a>
      <ul class="dropdown-menu">
        ...
      </ul>
    </li>
  </ul>
```

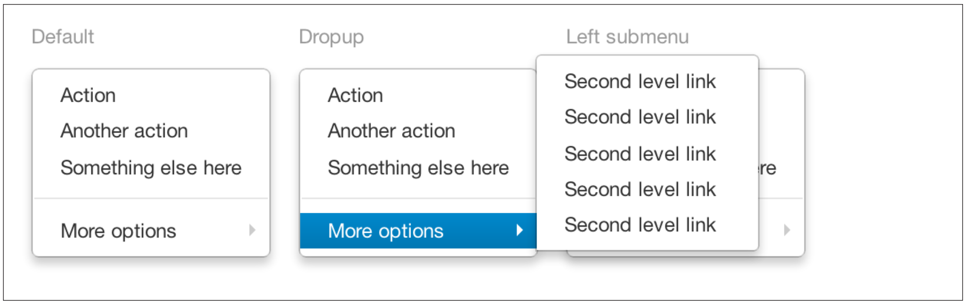


Figure 3-2. Dropdown menu and submenu

Button Groups

Button groups allow multiple buttons to be stacked together (see Figure 3-3). This is useful when you want to place items like alignment buttons together. To create a button group, simply wrap a series of anchors or buttons in a `<div>` that has `.btn-group` as a class:

```
<div class="btn-group">
  <button class="btn">1</button>
  <button class="btn">2</button>
  <button class="btn">3</button>
</div>
```

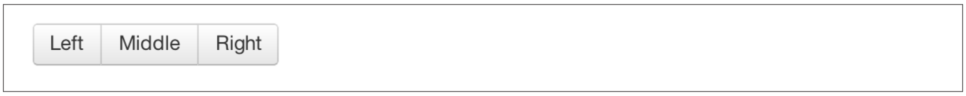


Figure 3-3. Left, middle, and right button group

If you have multiple button groups (see Figure 3-4) that you want to place on a single line, wrap multiple `.btn-group` classes with `.btn-toolbar`:

```
<div class="btn-toolbar">
  <div class="btn-group">
    <a class="btn" href="#"><i class="icon-align-left"></i></a>
    <a class="btn" href="#"><i class="icon-align-center"></i></a>
    <a class="btn" href="#"><i class="icon-align-right"></i></a>
    <a class="btn" href="#"><i class="icon-align-justify"></i></a>
  </div>
  <div class="btn-group">
    <a class="btn" href="#"><i class="icon-italic"></i></a>
    <a class="btn" href="#"><i class="icon-bold"></i></a>
    <a class="btn" href="#"><i class="icon-font"></i></a>
    <a class="btn" href="#"><i class="icon-text-height"></i></a>
    <a class="btn" href="#"><i class="icon-text-width"></i></a>
  </div>
  <div class="btn-group">
    <a class="btn" href="#"><i class="icon-indent-left"></i></a>
    <a class="btn" href="#"><i class="icon-indent-right"></i></a>
  </div>
</div>
```

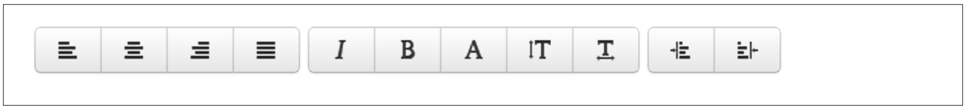


Figure 3-4. Button toolbar

For more information about using icons with buttons, follow the examples in Chapter 2.

To stack the buttons vertically (see Figure 3-5), add `.btn-group-vertical` to the `.btn-group` class:

```
<div class="btn-group btn-group-vertical">
  ...
</div>
```

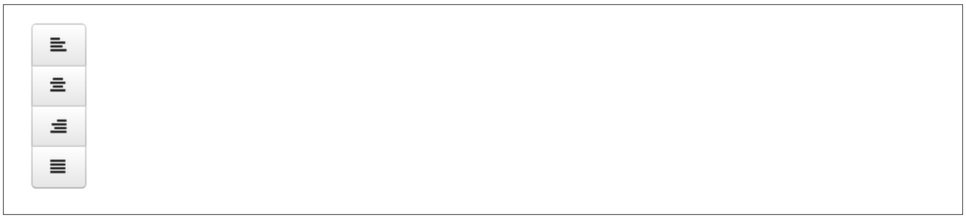


Figure 3-5. Vertical button group

Button Groups as Radio Buttons and Checkboxes

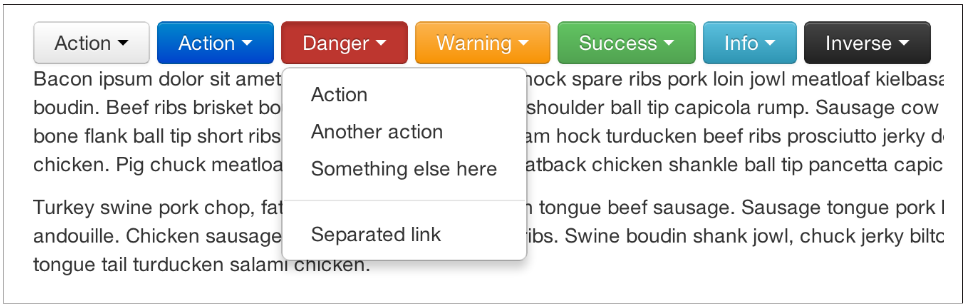
To have the checkboxes function as radio buttons, where only one option can be selected at a time, or checkboxes, where multiple options can be selected, you simply need to add some extra markup and then Bootstrap's JavaScript will provide the rest. This will be covered in detail in Chapter 4.

To use a button with a dropdown, it must be individually wrapped in its own `.btn-group` within a `btn-toolbar` for proper rendering.

Buttons with Dropdowns

To add a dropdown to a button (see Figure 3-6), simply wrap the button and dropdown menu in a `.btn-group`. You can also use `<span class="caret"></span>` to act as an indicator that the button is a dropdown:

```
<div class="btn-group">
  <button class="btn btn-danger">Danger</button>
  <button class="btn btn-danger dropdown-toggle" data-toggle="dropdown">
    <span class="caret"></span>
  </button>
  <ul class="dropdown-menu">
    <li><a href="#">Action</a></li>
    <li><a href="#">Another action</a></li>
    <li><a href="#">Something else here</a></li>
    <li class="divider"></li>
    <li><a href="#">Separated link</a></li>
  </ul>
</div>
```



Bacon ipsum dolor sit amet hock spare ribs pork loin jowl meatloaf kielbasa
boudin. Beef ribs brisket bo shoulder ball tip capicola rump. Sausage cow
bone flank ball tip short ribs am hock turducken beef ribs prosciutto jerky d
chicken. Pig chuck meatloa atback chicken shankle ball tip pancetta capic

Turkey swine pork chop, fat tongue beef sausage. Sausage tongue pork l
andouille. Chicken sausage ibs. Swine boudin shank jowl, chuck jerky biltc
tongue tail turducken salami chicken.

Figure 3-6. Button with a dropdown

You can use the dropdowns with any button size: `.btn-large`, `.btn`, `.btn-small`, or `.btn-mini`. Figure 3-7 shows several examples of different button sizes.

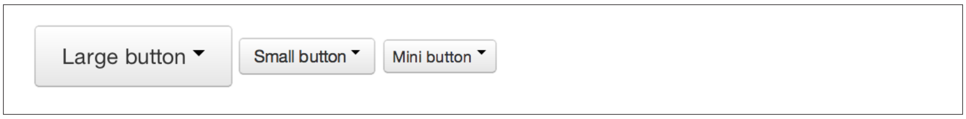
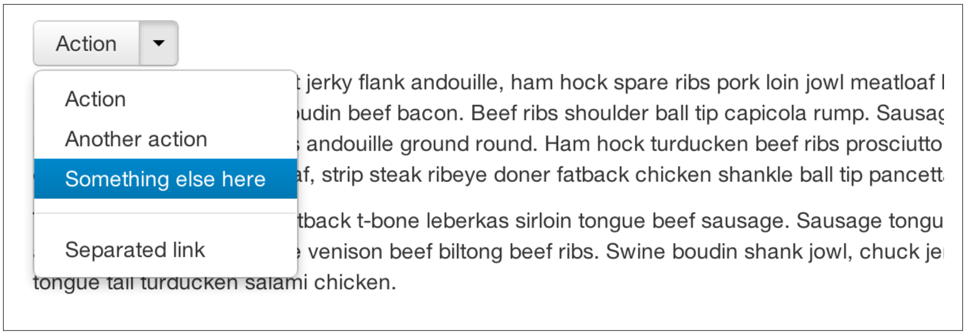


Figure 3-7. Button dropdown sizes

Split Button Dropdowns

Split button dropdowns (see Figure 3-8) use the same general style as the dropdown button but add a primary action along with the dropdown. Split buttons have the primary action on the left and a toggle on the right that displays the dropdown.



t jerky flank andouille, ham hock spare ribs pork loin jowl meatloaf l
udin beef bacon. Beef ribs shoulder ball tip capicola rump. Sausag
andouille ground round. Ham hock turducken beef ribs prosciutto
af, strip steak ribeye doner fatback chicken shankle ball tip pancett

tback t-bone leberkas sirloin tongue beef sausage. Sausage tongu
venison beef biltong beef ribs. Swine boudin shank jowl, chuck je
tongue tail turducken salami chicken.

Figure 3-8. Split button dropdown

Here's the code for a split button dropdown:

```
<div class="btn-group">
  <button class="btn">Action</button>
  <button class="btn dropdown-toggle" data-toggle="dropdown">
```

```
    <span class="caret"></span>
  </button>
  <ul class="dropdown-menu">
    <!-- dropdown menu links -->
  </ul>
</div>
```

Dropup Menus

Menus can also be built to drop up rather than down (see Figure 3-9). To make this change, simply add `.dropup` to the `.btn-group` container. To have the button pull up from the righthand side, add `.pull-right` to the `.dropdown-menu` (take note: the caret is now pointed up because the menu will be going up instead of down):

```
<div class="btn-group dropup">
  <button class="btn">Dropup</button>
  <button class="btn dropdown-toggle" data-toggle="dropdown">
    <span class="caret"></span>
  </button>
  <ul class="dropdown-menu">
    <!-- dropdown menu links -->
  </ul>
</div>
```

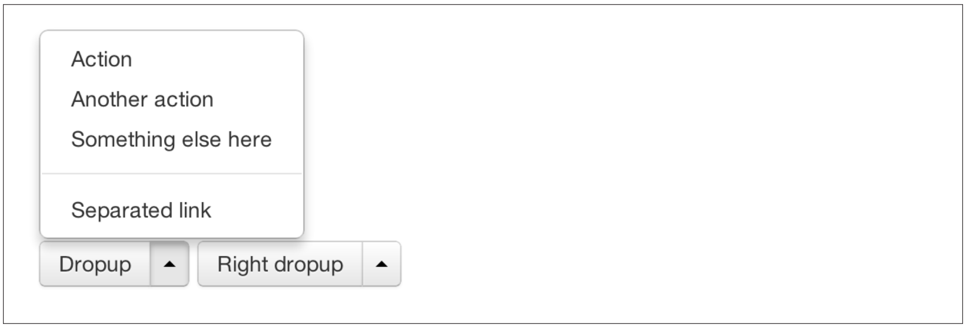


Figure 3-9. Dropup menu

Navigation Elements

Bootstrap provides a few different options for styling navigation elements. All of them share the same markup and base class, `.nav`.

Bootstrap also provides a helper class, `.active`. In principle, it generally adds distinction to the current element and sets it apart from the rest of the navigation elements. You can add this class to the home page links or to the links of the page that the user is currently on.

Tabular Navigation

To create a tabbed navigation menu (see Figure 3-10), start with a basic unordered list with the base class of `.nav` and add `.nav-tabs`:

```
<ul class="nav nav-tabs">
  <li class="active">
    <a href="#">Home</a>
  </li>
  <li><a href="#">Profile</a></li>
  <li><a href="#">Messages</a></li>
</ul>
```

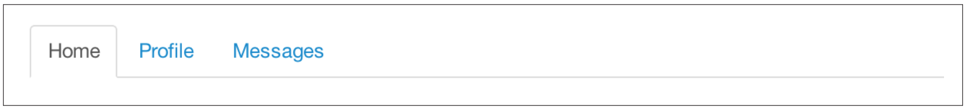


Figure 3-10. Tabbed navigation

Basic Pills Navigation

To turn the tabs into pills (see Figure 3-11), use `.nav-pills` instead of `.nav-tabs`:

```
<ul class="nav nav-pills">
  <li class="active">
    <a href="#">Home</a>
  </li>
  <li><a href="#">Profile</a></li>
  <li><a href="#">Messages</a></li>
</ul>
```

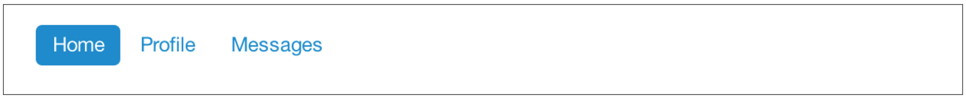


Figure 3-11. Tabbed navigation

Disabled class

For each of the `.nav` classes, if you add the `.disabled` class, it will create a gray link that also disables the `:hover` state (see Figure 3-12):

```
<ul class="nav nav-pills">
  ...
  <li class="disabled"><a href="#">Home</a></li>
  ...
</ul>
```

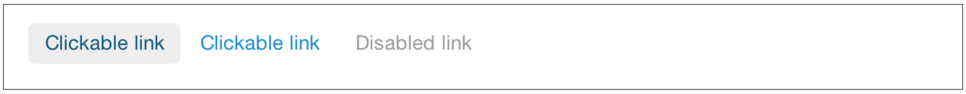


Figure 3-12. Disabled navigation

The link is still clickable unless the `href` is removed with JavaScript or some other method.

Stackable Navigation

Both tabs and pills are horizontal by default. To make them appear vertically stacked, just add the `.nav-stacked` class. See Figures 3-13 and 3-14 for examples of verticaly stacked tabs and pills.

Here's the code for stacked tabs:

```
<ul class="nav nav-tabs nav-stacked">
  ...
</ul>
```

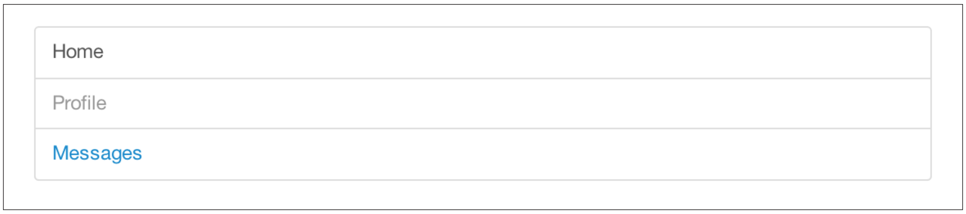


Figure 3-13. Stacked tabs

Here's the code for stacked pills:

```
<ul class="nav nav-pills nav-stacked">
  ...
</ul>
```

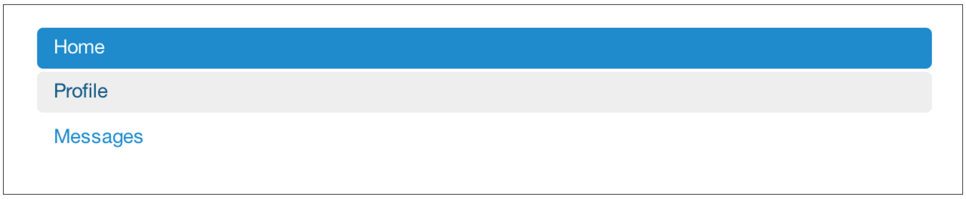


Figure 3-14. Stacked pills

Dropdowns

Navigation menus share a similar syntax with dropdown menus (see Figure 3-15). By default, you have a list item that has an anchor working in conjunction with some `data-` attributes to trigger an unordered list with a `.dropdown-menu` class:

```
<ul class="nav nav-tabs">
        <li class="dropdown">
                <a class="dropdown-toggle"
                        data-toggle="dropdown"
                        href="#">
                        Dropdown
                        <b class="caret"></b>
                </a>
                <ul class="dropdown-menu">
      <li><a href="#">Action</a></li>
      <li><a href="#">Another action</a></li>
      <li><a href="#">Something else here</a></li>
      <li class="divider"></li>
      <li><a href="#">Separated link</a></li>
    </ul>
        </li>
</ul>
```

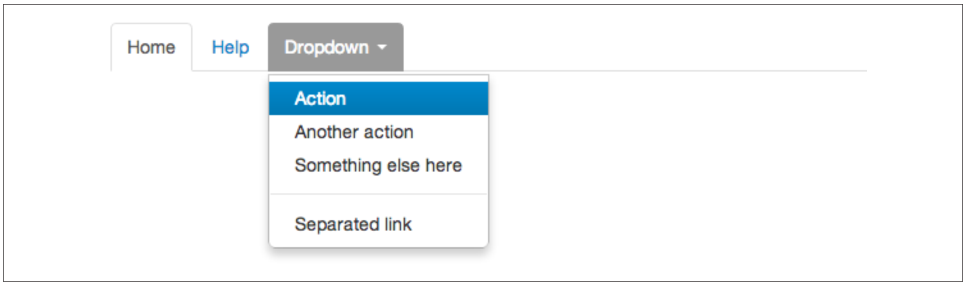


Figure 3-15. Tabbed navigation with a dropdown menu

To do the same thing with pills (Figure 3-16), simply swap the `.nav-tabs` class with `.nav-pills`:

```
<ul class="nav nav-pills">
        <li class="dropdown">
                <a class="dropdown-toggle" data-toggle="dropdown" href="#">
                        Dropdown
                        <b class="caret"></b>
                </a>
                <ul class="dropdown-menu">
                        <!--links-->
                </ul>
        </li>
</ul>
```

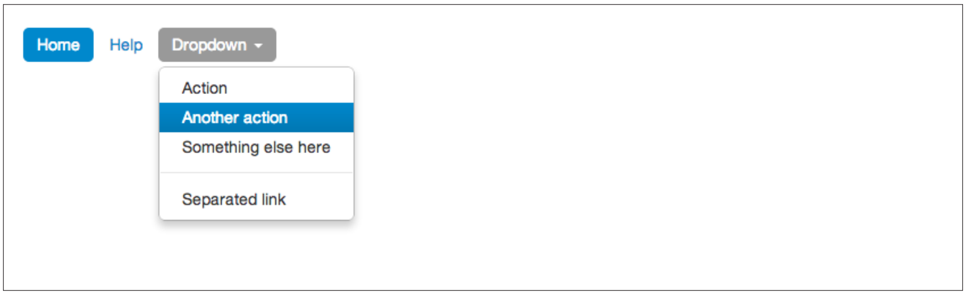


Figure 3-16. Pill navigation with dropdowns

Navigation Lists

Navigation lists are useful when you need to display a group of navigation links. This type of element is common when building admin interfaces. In the MAKE admin interface, for example, I have one of these on the sidebar of every page with quick links to common pages (see Figure 3-17). Bootstrap developers use a form of this for their documentation. Like all of the lists that we have discussed thus far, navigation lists are unordered lists with the `.nav` class. To give it its specific styling, we add the `.nav-list` class:

```
<ul class="nav nav-list">
        <li class="nav-header">List Header</li>
        <li class="active"><a href="/">Home</a></li>
        <li><a href="#">Library</a></li>
  <li><a href="#">Applications</a></li>
  <li class="nav-header">Another List Header</li>
  <li><a href="#">Profile</a></li>
        <li><a href="#">Settings</a></li>
  <li class="divider"></li>
  <li><a href="#">Help</a></li>
</ul>
```

Horizontal divider

To create a divider, much like an `<hr />`, use an empty `<li>` with a class of `.divider`:

```
<ul class="nav-menu">
    ...
        <li class="divider"></li>
        ....
</ul>
```

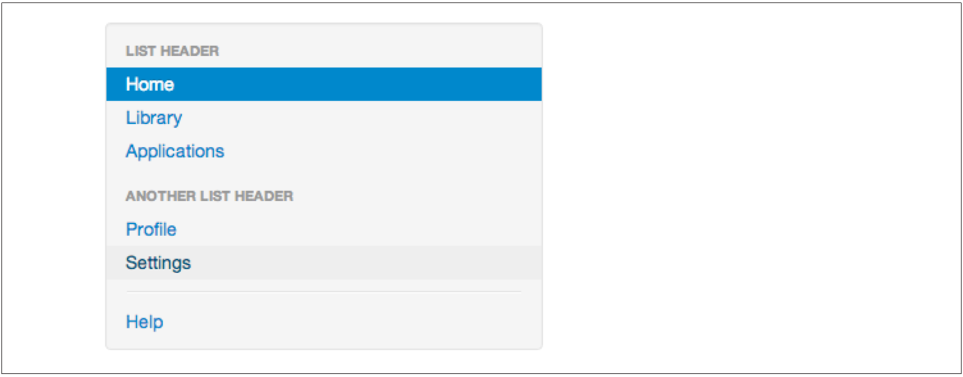
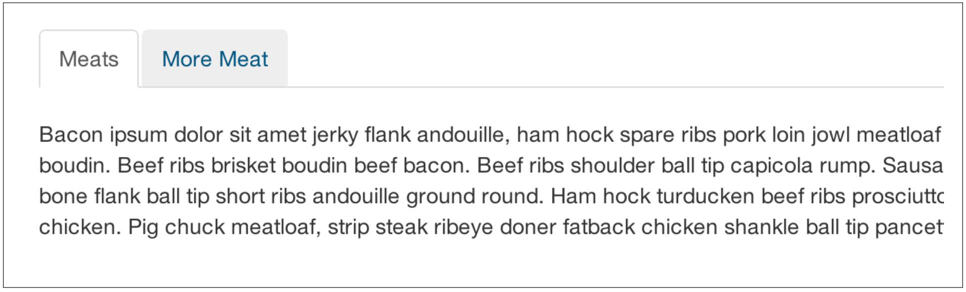


Figure 3-17. Navigation list

Tabbable Navigation

Not only can you create a tabbed navigation, but by using the JavaScript plugin, you can also add interaction by opening different windows of content (see Figure 3-18). To make navigation tabs, create a `.tab-pane` with a unique ID for every tab, and then wrap them in `.tab-content`:

```
<div class="tabbable">
        <ul class="nav nav-tabs">
           <li class="active"><a href="#tab1" data-toggle="tab">Meats</a></li>
           <li><a href="#tab2" data-toggle="tab">More Meat</a></li>
        </ul>
        <div class="tab-content">
           <div class="tab-pane active" id="tab1">
                   <p>Bacon ipsum dolor sit amet jerky flank...</p>
           </div>
           <div class="tab-pane" id="tab2">
                   <p>Beef ribs, turducken ham hock...</p>
           </div>
        </div>
</div>
```



Bacon ipsum dolor sit amet jerky flank andouille, ham hock spare ribs pork loin jowl meatloaf boudin. Beef ribs brisket boudin beef bacon. Beef ribs shoulder ball tip capicola rump. Sausa bone flank ball tip short ribs andouille ground round. Ham hock turducken beef ribs prosciuttc chicken. Pig chuck meatloaf, strip steak ribeye doner fatback chicken shankle ball tip pancet

Figure 3-18. Tabbable navigation example

If you want to make the tabs fade when switching, add `.fade` to each `.tab-pane`.

Tab position

The tabs are fully positionable; you can have them above, below, or on the sides of the content (see Figure 3-19).

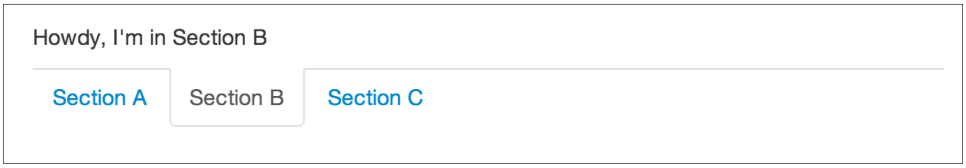


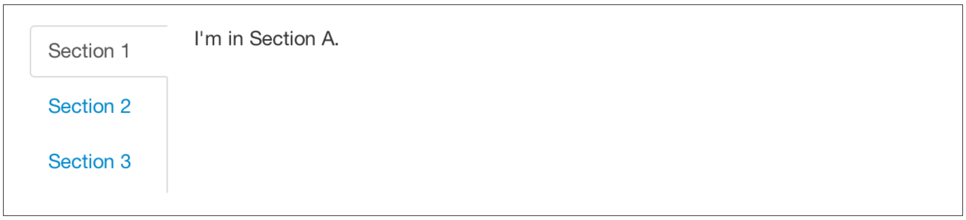
Figure 3-19. Bottom tabs

Here's the code for positioning tabs:

```
<div class="tabbable tabs-below">
  <div class="tab-content">
    <div class="tab-pane active" id="tab1">
      <p>I'm in Section A.</p>
    </div>
    <div class="tab-pane" id="tab2">
      <p>I'm in Section B.</p>
    </div>
    <div class="tab-pane" id="tab3">
      <p>I'm in Section C.</p>
    </div>
  </div>
  <ul class="nav nav-tabs">
    <li class="active"><a href="#tab1" data-toggle="tab">Section A</a></li>
    <li><a href="#tab2" data-toggle="tab">Section B</a></li>
    <li><a href="#tab3" data-toggle="tab">Section C</a></li>
  </ul>
</div>
```

Tabs on the left (see Figure 3-20) get the `.tabs-left` class. For this, you need to swap the tab content and the tabs:

```
<div class="tabbable tabs-left">
  <ul class="nav nav-tabs">
    <li class="active"><a href="#tab1" data-toggle="tab">Section A</a></li>
    <li><a href="#tab2" data-toggle="tab">Section B</a></li>
    <li><a href="#tab3" data-toggle="tab">Section C</a></li>
  </ul>
  <div class="tab-content">
    <div class="tab-pane active" id="tab1">
      <p>I'm in Section A.</p>
    </div>
    <div class="tab-pane" id="tab2">
      <p>I'm in Section B.</p>
    </div>
    <div class="tab-pane" id="tab3">
      <p>I'm in Section C.</p>
    </div>
  </div>
</div>
```



I'm in Section A.

Figure 3-20. Left tabs

Tabs on the right get the `.tabs-right` class (see Figure 3-21):

```
<div class="tabbable tabs-right">
  <ul class="nav nav-tabs">
    <li class="active"><a href="#tab1" data-toggle="tab">Section A</a></li>
    <li><a href="#tab2" data-toggle="tab">Section B</a></li>
    <li><a href="#tab3" data-toggle="tab">Section C</a></li>
  </ul>
  <div class="tab-content">
    <div class="tab-pane active" id="tab1">
      <p>I'm in section A.</p>
    </div>
    <div class="tab-pane" id="tab2">
      <p>I'm in section B.</p>
    </div>
    <div class="tab-pane" id="tab3">
      <p>I'm in section C.</p>
    </div>
  </div>
</div>
```

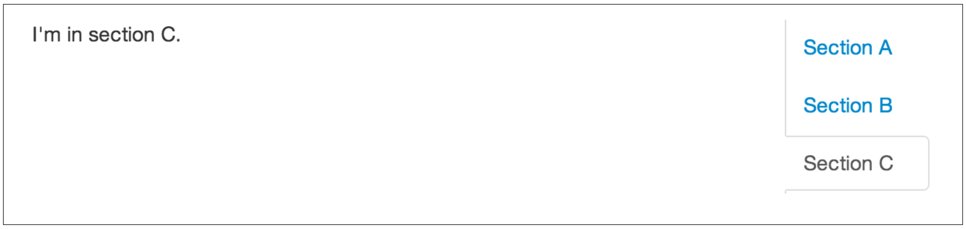

I'm in section C.

Figure 3-21. Right tabs

As a footnote to the tabbable elements, you can use the markup here to control a variety of things that are perhaps outside of the scope of the default usage mechanism. On MAKE's site, I use this to control the navigation and subnavigation. When you click on the navigation menu, the subnavigation changes and shows different links.

Navbar

The navbar is a nice feature, and is one of the prominent features of Bootstrap sites (see Figure 3-22). At its core, the navbar includes styling for site names and basic navigation. It can later be extended by adding form-specific controls and specialized dropdowns. To be sure that the navbar is constrained to the width of the content of the page, either place it inside of a `.span12` or the `.container` class:

```
<div class="navbar">
  <div class="navbar-inner">
    <a class="brand" href="#">Title</a>
    <ul class="nav">
      <li class="active"><a href="#">Home</a></li>
      <li><a href="#">Link</a></li>
      <li><a href="#">Link</a></li>
    </ul>
  </div>
</div>
```

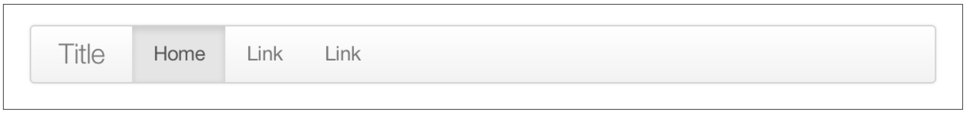


Figure 3-22. Basic navbar

Note the `.brand` class in the code. This will give the text a lighter `font-weight` and slightly larger size.

```
<a class="brand" href="#">Project name</a>
```

Navbar Links

To add links to the navbar (see Figure 3-23), simply add an unordered list with a class of `.nav`. If you want to add a divider to your links, you can do that by adding an empty list item with a class of `.divider-vertical`:

```
<ul class="nav">
  <li class="active"><a href="#">Home</a></li>
  <li><a href="#">First Link</a></li>
  <li><a href="#">Second Link</a></li>
```

```
    <li class="divider-vertical"></li>
    <li><a href="#">Third Link</a></li>
  </ul>
```

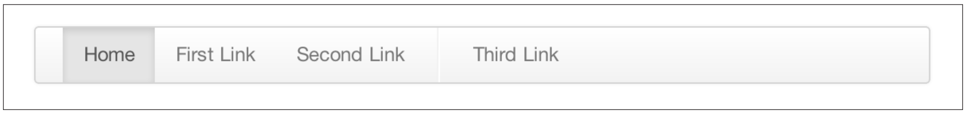


Figure 3-23. Nav links

Forms

Instead of using the default class-based forms from Chapter 2, forms that are in the navbar use the `.navbar-form` class. This ensures that the form's margins are properly set and match the nav stylings (see Figure 3-24). Of note, `.pull-left` and `.pull-right` helper classes may help move the form into the proper position:

```
<form class="navbar-form pull-left">
        <input type="text" class="span2" id="fname">
        <button type="submit" class="btn">
</form>
```

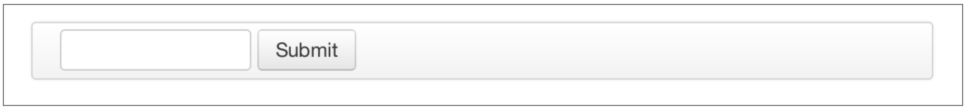


Figure 3-24. Default navbar form

To add rounded corners (see Figure 3-25), as seen in the search inputs of iOS devices, use the `.navbar-search` class instead of the `.navbar-form`:

```
<form class="navbar-search"  accept-charset="utf-8">
        <input type="text" class="search-query" placeholder="Search">
</form>
```

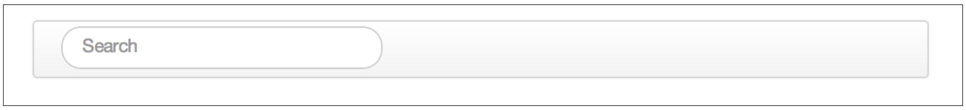


Figure 3-25. Navbar search input

Navbar Menu Variations

The Bootstrap navbar can be dynamic in its positioning. By default, it is a block-level element that takes its positioning based on its placement in the HTML. With a few helper classes, you can place it either on the top or bottom of the page, or you can make it scroll statically with the page.

Fixed top navbar

If you want the navbar fixed to the top, add `.navbar-fixed-top` to the `.navbar` class. To prevent the navbar from sitting on top of other content in the body of the page, add at least 40 pixels of padding to the `<body>` tag:

```
<div class="navbar navbar-fixed-top">
  <div class="navbar-inner">
    <a class="brand" href="#">Title</a>
    <ul class="nav">
      <li class="active"><a href="#">Home</a></li>
      <li><a href="#">Link</a></li>
      <li><a href="#">Link</a></li>
    </ul>
  </div>
</div>
```

Fixed bottom navbar

To affix the navbar to the bottom of the page, simply add the `.fixed-navbar-bottom` class to the navbar. Once again, to prevent overlap, add at least 40 pixels of padding to the `<body>` tag:

```
<div class="navbar navbar-fixed-bottom">
 <div class="navbar-inner">
    <a class="brand" href="#">Title</a>
    <ul class="nav">
      <li class="active"><a href="#">Home</a></li>
      <li><a href="#">Link</a></li>
      <li><a href="#">Link</a></li>
    </ul>
  </div>
</div>
```

Static top navbar

To create a navbar that scrolls with the page, add the `.navbar-static-top` class. This class does not require adding the padding to the `<body>`:

```
<div class="navbar navbar-static-top">
  <div class="navbar-inner">
    <a class="brand" href="#">Title</a>
    <ul class="nav">
      <li class="active"><a href="#">Home</a></li>
      <li><a href="#">Link</a></li>
      <li><a href="#">Link</a></li>
    </ul>
    </div>
  </div>
```

Responsive navbar

Like the rest of Bootstrap, the navbar can be totally responsive as shown in Figure 3-26. To add the responsive features, the content that you want to be collapsed needs to be wrapped in a `<div>` with `.nav-collapse.collapse` as a class. The collapsing nature is tripped by a button that has a the class of `.btn-navbar` and then features two `data-` elements. The first, `data-toggle`, is used to tell the JavaScript what to do with the button, and the second, `data-target`, indicates which element to toggle. Three `<spans>` with a class of `.icon-bar` create what I like to call the hamburger button. This will toggle the elements that are in the `.nav-collapse` `<div>`. For this feature to work, the *bootstrap-responsive.css* and either the *collapse.js* or the full *bootstrap.js* files must be included.

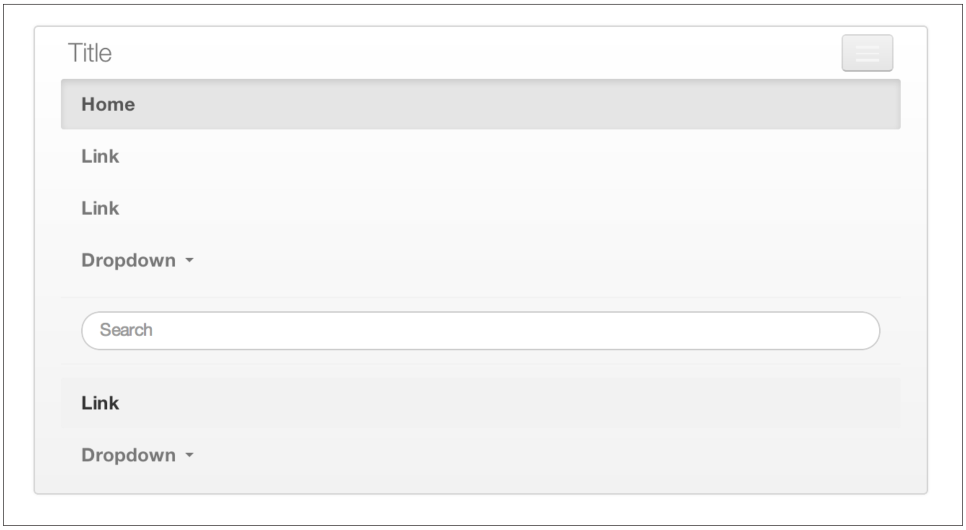


Figure 3-26. Responsive navbar

Use the following code to create a responsive navbar:

```
<div class="header">
  <div class="navbar-inner">
    <div class="container">
      <a class="btn btn-navbar" data-toggle="collapse"
      data-target=".nav-collapse">
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </a>
```

```
          <!-- Leave the brand out if you want it to be shown when other elements
          are collapsed... -->
          <a href="#" class="brand">Project Name</a>

          <!-- Everything that you want collapsed, should be added to the collapse
          div. -->
          <div class="nav-collapse collapse">
            <!-- .nav, .navbar-search etc... -->
          </div>

        </div>
      </div>
    </div>
```

Inverted navbar

To create an inverted navbar with a black background and white text as shown in Figure 3-27, simply add `.navbar-inverse` to the `.navbar` class:

```
<div class="navbar navbar-inverse">
        ...
</div>
```

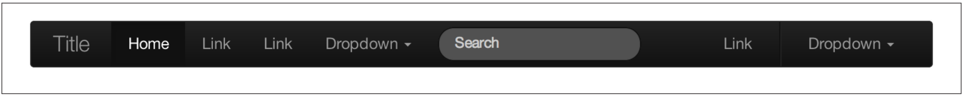


Figure 3-27. Inverted navbar

Breadcrumbs

Breadcrumbs are a great way to show hierarchy-based information for a site (see Figure 3-28). In the case of blogs, breadcrumbs can show the dates of publishing, categories, or tags. A breadcrumb in Bootstrap is simply an unordered list with a class of `.breadcrumb`. There is a also a helper class of `.divider` that mutes the colors and makes the text a little smaller. You can use forward slashes, arrows, or any divided that you choose. Note that the divider in the breadcrumbs has a slightly different markup than the navbar example.

The following code uses the class `.breadcrumb`:

```
<ul class="breadcrumb">
    <li><a href="#">Home</a> <span class="divider">/</span></li>
        <li><a href="#">2012</a> <span class="divider">/</span></li>
        <li><a href="#">December</a> <span class="divider">/</span></li>
        <li><a href="#">5</a></li>
</ul>
```

```
<ul class="breadcrumb">
  <li><a href="#">Home</a> <span class="divider">&rarr;</span></li>
  <li><a href="#">Dinner Menu</a> <span class="divider">&rarr;</span></li>
  <li><a href="#">Specials</a> <span class="divider">&rarr;</span></li>
  <li><a href="#">Steaks</a></li>
</ul>

<ul class="breadcrumb">
  <li><a href="#">Home</a> <span class="divider">&raquo;</span></li>
  <li><a href="#">Electronics</a> <span class="divider">&raquo;</span></li>
  <li><a href="#">Raspberry Pi</a></li>
</ul>
```

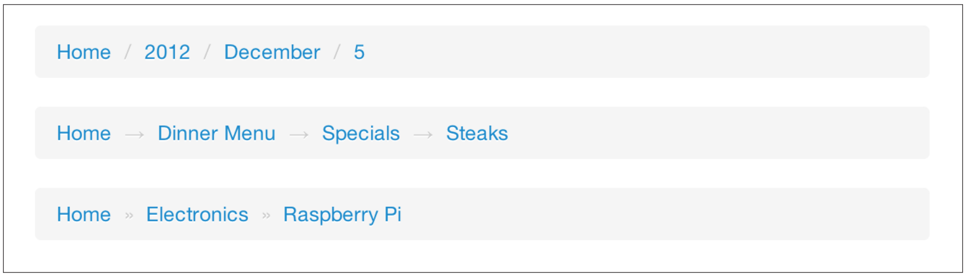

Home / 2012 / December / 5
Home → Dinner Menu → Specials → Steaks
Home » Electronics » Raspberry Pi

Figure 3-28. Breadcrumb

Pagination

Bootstrap handles pagination like a lot of other interface elements, an unordered list, with wrapper a `<div>` that has a specific class that identifies the element. In the basic form, adding `.pagination` to the parent `<div>` creates a row of bordered links. Each of the list items can be additionally styled by using the `.disabled` or `.active` class. See Figures 3-29 and 3-30 for examples of this.

Here's the code for basic pagination:

```
<div class="pagination">
  <ul>
    <li><a href="#">&laquo;</a></li>
    <li><a href="#">1</a></li>
    <li><a href="#">2</a></li>
    <li><a href="#">3</a></li>
    <li><a href="#">4</a></li>
    <li><a href="#">5</a></li>
    <li><a href="#">&raquo;</a></li>
  </ul>
</div>
```

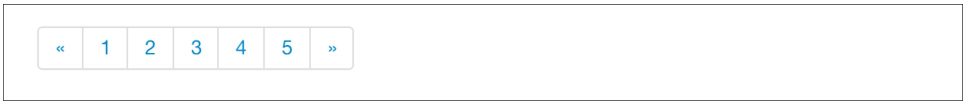


Figure 3-29. Basic pagination

And here's the code for pagination using helper classes:

```
<div class="pagination pagination-centered">
  <ul>
    <li class="disabled"><a href="#">«</a></li>
    <li class="active"><a href="#">1</a></li>
    <li><a href="#">2</a></li>
    <li><a href="#">3</a></li>
    <li><a href="#">4</a></li>
    <li><a href="#">5</a></li>
    <li><a href="#">»</a></li>
  </ul>
</div>
```

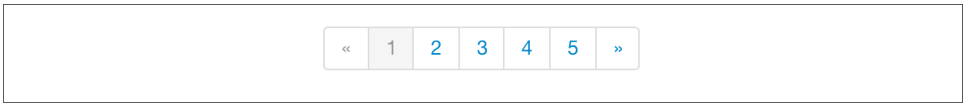


Figure 3-30. Pagination with helper classes

In addition to the `.active` and `.disabled` classes for list items, you can add `.pagination-centered` to the parent `<div>`. This will center the contents of the `<div>`. If you want the items right-aligned in the `<div>`, add `.pagination-right`. For sizing, in addition to the normal size, there are three other sizes that can be applied by adding a class to the wrapper `<div>`: `.pagination-large`, `.pagination-small`, and `.pagination-mini` (see Figure 3-31):

```
<div class="pagination pagination-large">
  <ul>
    ...
  </ul>
</div>
<div class="pagination">
  <ul>
    ...
  </ul>
</div>
<div class="pagination pagination-small">
  <ul>
    ...
  </ul>
</div>
<div class="pagination pagination-mini">
```

```
    <ul>
      ...
    </ul>
  </div>
```

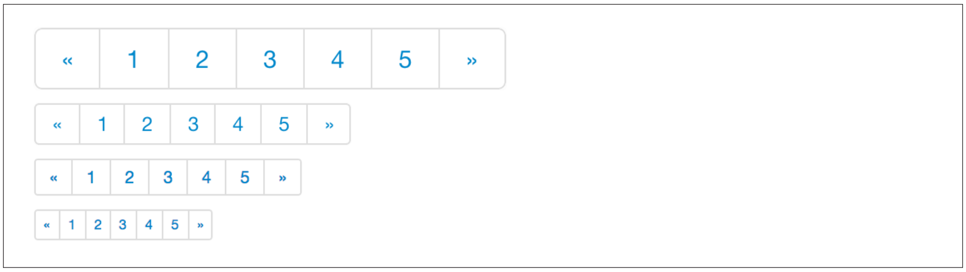


Figure 3-31. Pagination sizes

Pager

If you need to create simple pagination links that go beyond text, the pager can work quite well. Like the pagination links, the markup is an unordered list that sheds the wrapper `<div>`. By default, the links are centered (see Figure 3-32).

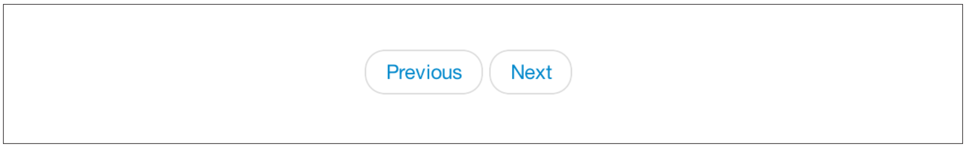


Figure 3-32. Basic pager

The following is the code for a basic pager:

```
<ul class="pager">
  <li><a href="#">Previous</a></li>
  <li><a href="#">Next</a></li>
</ul>
```

To left- or right-align the links, you just need to add the `.previous` and `.next` class as to the list items (see Figure 3-33). Also, like `.pagination` in Figure 3-31, you can add the `.disabled` class for a muted look.

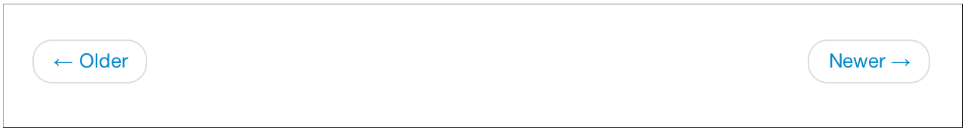


Figure 3-33. Aligned page links

The following is the code for aligning page links:

```
<ul class="pager">
  <li class="previous">
    <a href="#">&larr; Older</a>
  </li>
  <li class="next">
    <a href="#">Newer &rarr;</a>
  </li>
</ul>
```

Labels

Labels are great for offering counts, tips, or other markup for pages. They're another of my favorite little Bootstrap touches. Figure 3-34 shows some labels that can be used.

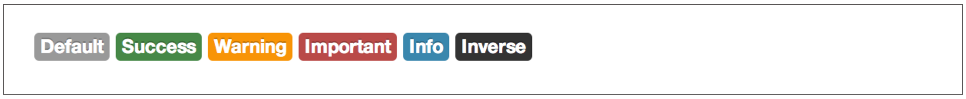


Figure 3-34. Labels

Here's the code to use these labels:

```
<span class="label">Default</span>
<span class="label label-success">Success</span>
<span class="label label-warning">Warning</span>
<span class="label label-important">Important</span>
<span class="label label-info">Info</span>
<span class="label label-inverse">Inverse</span>
```

Badges

Badges are similar to labels; the primary difference is that the corners are more rounded. The colors of badges reflect the same classes as labels (see Figure 3-35).

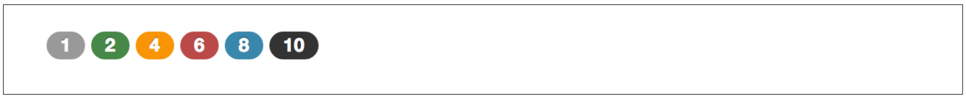


Figure 3-35. Badges

The following code shows how to use badges:

```
<span class="badge">1</span>
<span class="badge badge-success">2</span>
<span class="badge badge-warning">4</span>
<span class="badge badge-important">6</span>
```

```
<span class="badge badge-info">8</span>
<span class="badge badge-inverse">10</span>
```

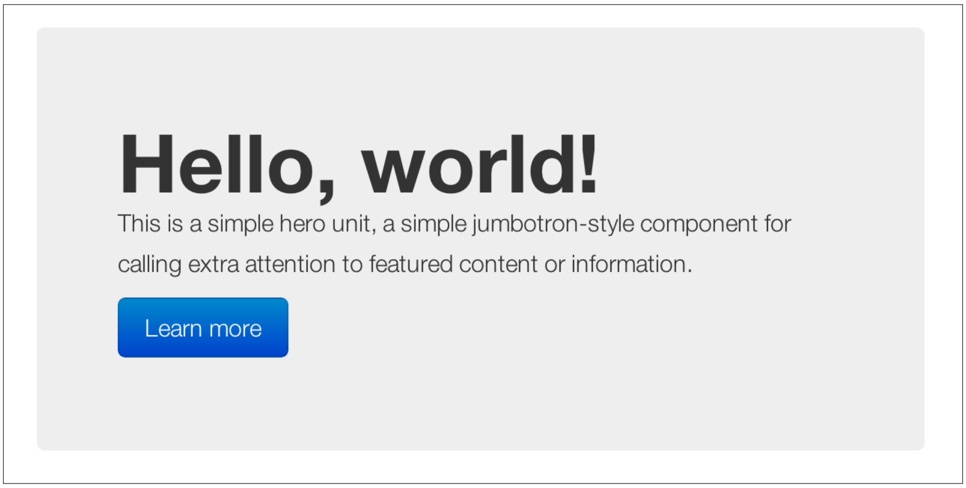
Typographic Elements

In addition to buttons, labels, forms, tables, and tabs, Bootstrap has a few more elements for basic page layout.

Hero Unit

The hero unit is a large content area that increases the size of headings and adds a lot of margin for landing page content (see Figure 3-36). To use the hero unit, simply create a container `<div>` with the class of `.hero-unit`. In addition to a larger `<h1>`, the `font-weight` is reduced to 200 :

```
<div class="hero-unit">
  <h1>Hello, World!</h1>
  <p>This is a simple hero unit, a simple jumbotron-style component for calling
  extra attention to featured content or information.</p>
  <p><a class="btn btn-primary btn-large">Learn more</a></p>
</div>
```


Hello, world!

This is a simple hero unit, a simple jumbotron-style component for calling extra attention to featured content or information.

Figure 3-36. Hero unit

Page Header

The page header (see Figure 3-37) is a nice little feature to add appropriate spacing around the headings on a page. This is particularly helpful on a blog archive page where you may have several post titles and need a way to add distinction to each of them. To use a page header, wrap your heading in a `<div>` with a class of `.page-header`:

```
<div class="page-header">
  <h1>Example page header <small>Subtext for header</small></h1>
</div>
```

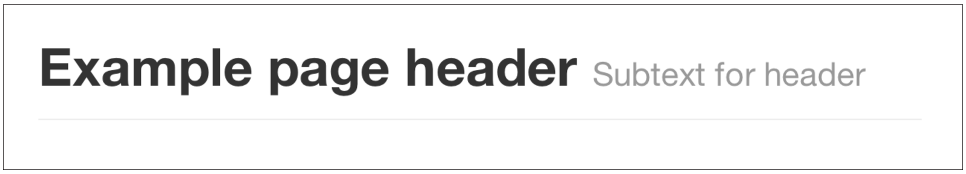


Figure 3-37. Page header

Thumbnails

A lot of sites need a way to lay out images in a grid, and Bootstrap has an easy way to do this. To create a thumbnail, add an `<a>` tag with the class of `.thumbnail` around an image. This adds four pixels of padding and a gray border (see Figure 3-38). On hover, an animated glow outlines the image.

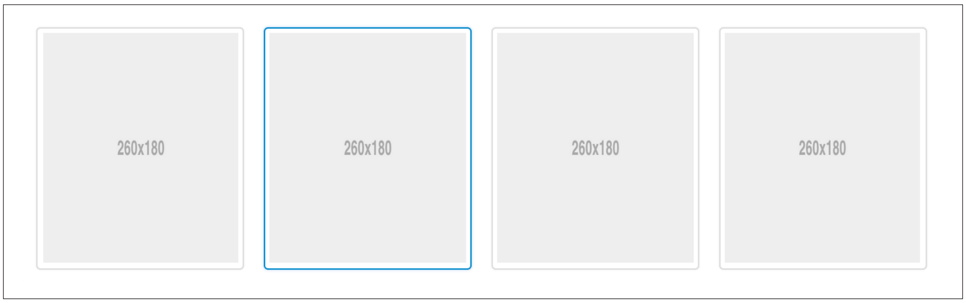


Figure 3-38. Basic thumbnail

Use the following code to create a thumbnail:

```
<a href="#" class="thumbnail">
  <img alt="Kittens!" style="" src="http://placekitten.com/300/250">
</a>
```

Now that you have your basic thumbnail, you can add headings, buttons, and more as shown in Figure 3-39; just change the `<a>` tag that has a class of `.thumbnail` to a `<div>`. Inside of that `<div>`, you can add anything you need. Since this is a `<div>`, we can use the default span-based naming convention for sizing. If you want to group multiple images, place them in an unordered list, and each list item will be floated to the left.

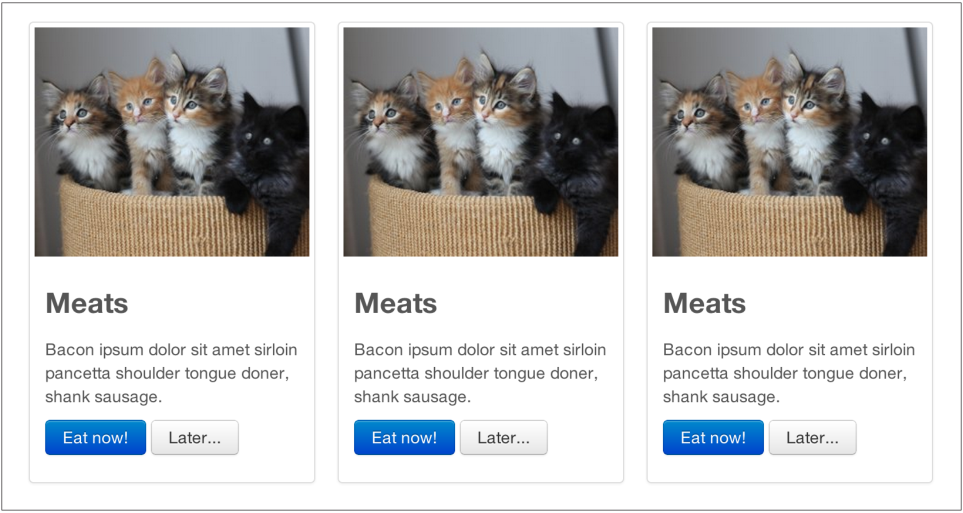

Meats
Bacon ipsum dolor sit amet sirloin pancetta shoulder tongue doner, shank sausage.
Eat now!
Later...

Figure 3-39. Extended thumbnail

The following code shows how to extend and add more to the thumbnail:

```
<ul class="thumbnails">
  <li class="span4">
    <div class="thumbnail">
      <img data-src="holder.js/300x200" alt="300x200" style="">
      <div class="caption">
        <h3>Meats</h3>
        <p>Bacon ipsum dolor sit amet sirloin pancetta shoulder tongue doner,
           shank sausage.</p>
        <p><a href="#" class="btn btn-primary">Eat now!</a> <a href="#"
           class="btn">Later...</a></p>
      </div>
    </div>
  </li>
  <li class="span4">
    ...
  </li>
</ul>
```

Alerts

Like the modals that will be described in Chapter 4, alerts provide a way to style messages to the user (see Figure 3-40). The default alert is added by creating a wrapper `<div>` and adding a class of `.alert`:

```
<div class="alert">
    <a href="#" class="close" data-dismiss="alert">&times;</a>
    <strong>Warning!</strong> Not to be alarmist, but you have now been alerted.
</div>
```

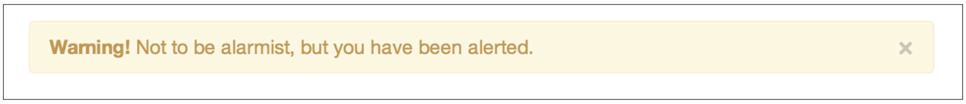

Warning! Not to be alarmist, but you have been alerted. ×

Figure 3-40. Basic alert

The `.alert` uses the alerts jQuery plugin that is discussed in Chapter 4. To close the alert, you can use a button that contains the `data-dismiss="alert"` attribute. Mobile Safari and Mobile Opera browsers require an `href="#"` to close.

If you have a longer message in your alert, you can use the `.alert-block` class. This provides a little more padding above and below the content contained in the alert, which is particularly useful for multi-page lines of content (see Figure 3-41).

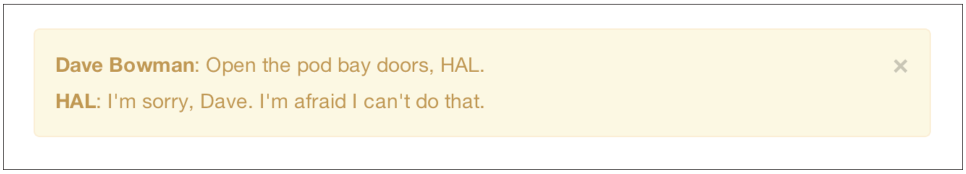
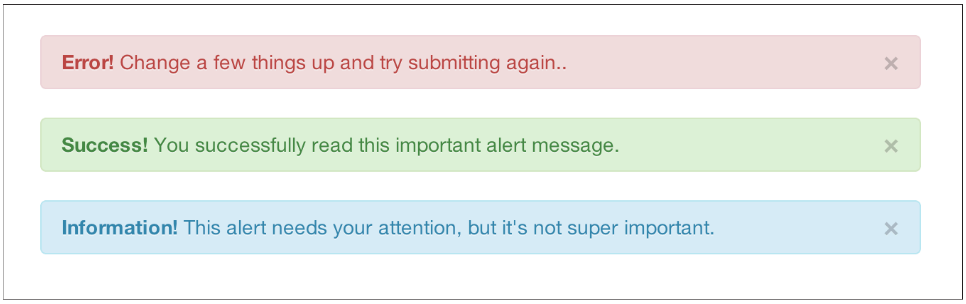

Dave Bowman: Open the pod bay doors, HAL. ×
HAL: I'm sorry, Dave. I'm afraid I can't do that.

Figure 3-41. Alert block

There are also three other color options as shown in Figure 3-42 to help provide a more semantic method for the alert. They are added by using either `.alert-error`, `.alert-success`, or `.alert-info`.


Error! Change a few things up and try submitting again.. ×

Success! You successfully read this important alert message. ×

Information! This alert needs your attention, but it's not super important. ×

Figure 3-42. Alert color options

Progress Bars

The purpose of progress bars is to show that assets are loading, in progress, or that there is action taking place regarding elements on the page. Personally, I think that these elements are just an exercise in markup and have little purpose beyond that in the

Bootstrap framework. That being said, among the thousands of people using Bootstrap, there are likely a few outliers who have a good reason for building progress bars. By nature, these are static elements that need some sort of JavaScript method to provide any interaction.

The default progress bar has a light gray background and a blue progress bar as shown in Figure 3-43. To create it, add a `<div>` with a class of `.progress`. Inside, add an empty `<div>` with a class of `.bar`. Add a style attribute with the width expressed as a percentage. I added `style="60%";` to indicate that the progress bar was at 60%:

```
<div class="progress">
  <div class="bar" style="width: 60%;"></div>
</div>
```

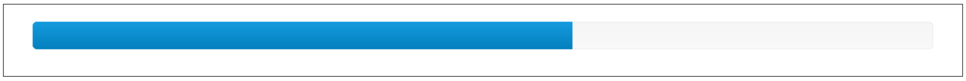


Figure 3-43. Default progress bar

To create a striped progress bar (see Figure 3-44),[1] just add `.progress-striped` to the container `<div>`:

```
<div class="progress progress-striped">
  <div class="bar" style="width: 20%;"></div>
</div>
```

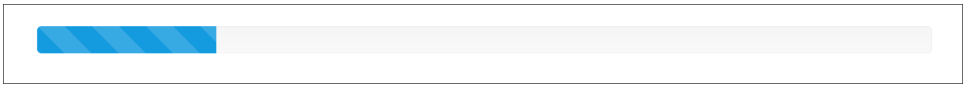


Figure 3-44. Striped progress bar

Like the striped version of the progress bar, you can animate the stripes (see Figure 3-45 for a static image of this), making it look like a blue light special barbershop pole.

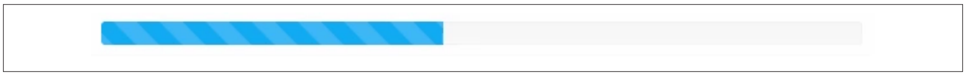


Figure 3-45. Animated progress bar

Here's the code to animate the progress bar:

1. Striped progress bars are not available in Internet Explorer 7 and 8.

```
<div class="progress progress-striped active">
  <div class="bar" style="width: 40%;"></div>
</div>
```

In addition to the blue progress bar, there are options for green, yellow, and red using the `.bar-success`, `.bar-warning`, and `.bar-danger` classes. Progress bars can be stacked (see Figure 3-46), indicating a graph of sorts by adding multiple elements together using this code:

```
<div class="progress">
  <div class="bar bar-success" style="width: 35%;"></div>
  <div class="bar bar-warning" style="width: 20%;"></div>
  <div class="bar bar-danger" style="width: 10%;"></div>
</div>
```

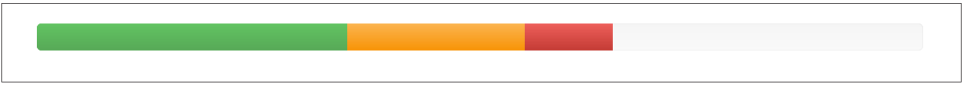
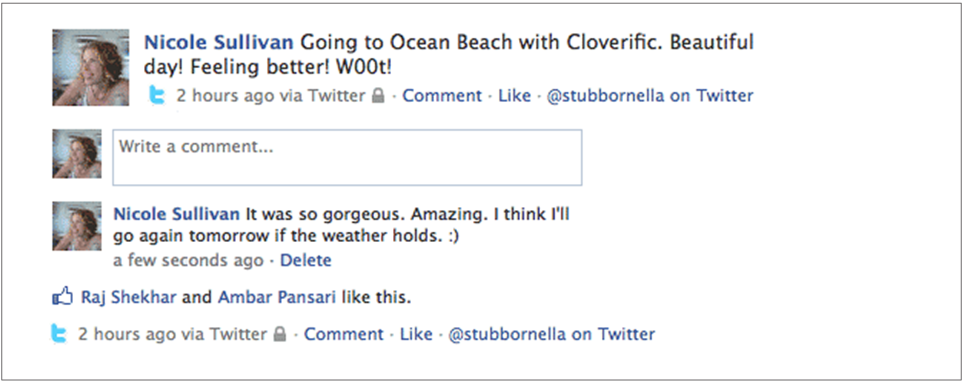


Figure 3-46. Stacked progress bar

Media Object

When you look at social sites like Facebook, Twitter, and others, and strip away some of the formatting from timelines, you will see the media object (see Figure 3-47). Driven by the Bootstrap community and based on principles from the oocss community, the goal of the media object is to make the code for developing these blocks of information drastically shorter. Nicole Sullivan-Hass shares a few elements of the media object similar to Bootstrap's on her site. The media object is designed to literally save hundreds of lines of code, making it easy to customize.



Nicole Sullivan Going to Ocean Beach with Cloverific. Beautiful day! Feeling better! W00t!
2 hours ago via Twitter · Comment · Like · @stubbornella on Twitter

Write a comment...

Nicole Sullivan It was so gorgeous. Amazing. I think I'll go again tomorrow if the weather holds. :)
a few seconds ago · Delete

Raj Shekhar and Ambar Pansari like this.

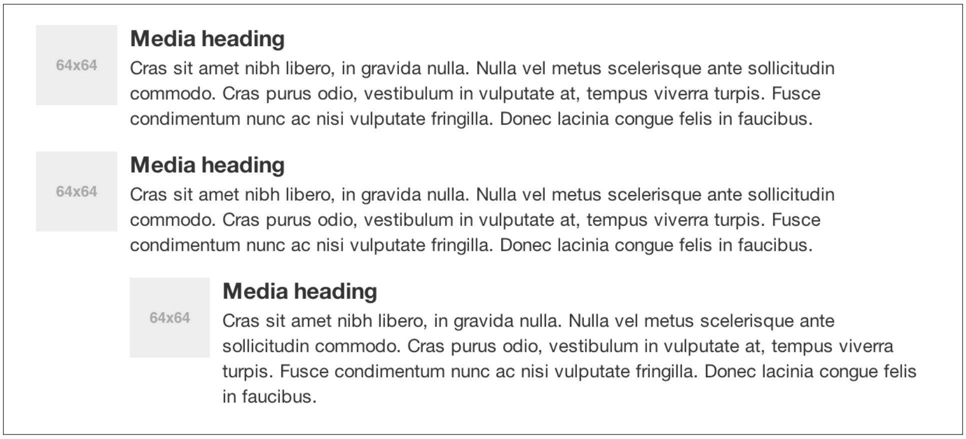
2 hours ago via Twitter · Comment · Like · @stubbornella on Twitter

Figure 3-47. Media object

Bootstrap leaves the design and formatting to you but provides a simple way to get going. Like a lot of other tools in Bootstrap, the goal of media objects (light markup, easy extendability) is achieved by applying classes to some simple markup. There are two forms to the media object: `.media` and `.media-list`. Figure 3-48 shows the former form. If you are preparing a list where the items will be part of an unordered list, use `.media-list`. If you are using only just `<div>` elements, use the `.media` object:

```
<div class="media">
  <a class="pull-left" href="#">
    <img class="media-object" data-src="holder.js/64x64">
  </a>
  <div class="media-body">
    <h4 class="media-heading">Media heading</h4>
    <p>...</p>

    <!-- Nested media object -->
    <div class="media">
      ...
    </div>
  </div>
</div>
```



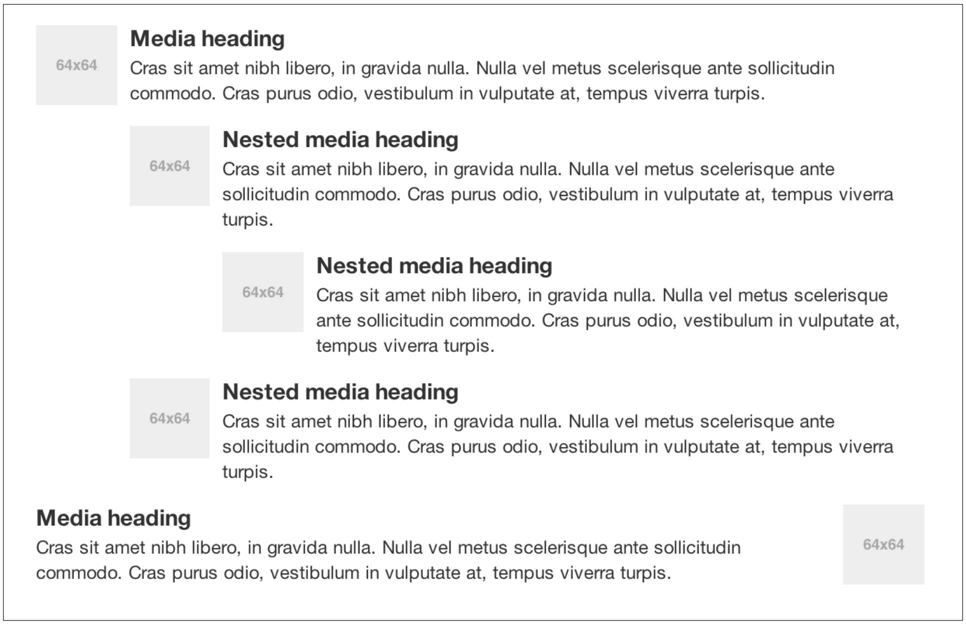
Media heading
Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis. Fusce condimentum nunc ac nisi vulputate fringilla. Donec lacinia congue felis in faucibus.

Media heading
Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis. Fusce condimentum nunc ac nisi vulputate fringilla. Donec lacinia congue felis in faucibus.

Media heading
Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis. Fusce condimentum nunc ac nisi vulputate fringilla. Donec lacinia congue felis in faucibus.

Figure 3-48. Default media object

To use media list (shown in Figure 3-49), change the container `<div>` to an `<ul>` and add the class `.media-list`. Since you can nest media objects, it is handy to markup for comments or other lists.



Media heading

Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis.

Nested media heading

Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis.

Nested media heading

Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis.

Nested media heading

Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis.

Media heading

Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis.

Figure 3-49. Media list example

The following code creates a media list:

```
<ul class="media-list">
  <li class="media">
    <a class="pull-left" href="#">
      <img class="media-object" data-src="holder.js/64x64">
    </a>
    <div class="media-body">
      <h4 class="media-heading">Media heading</h4>
      <p>...</p>
      ...

      <!-- Nested media object -->
      <div class="media">
        ...
     </div>
    </div>
  </li>
</ul>
```

Miscellaneous

There are a few more Bootstrap components that we have yet to cover in this chapter. Some of these components are layout-based, and a few are production-based helper classes. The first among these are the wells.

Wells

A well is a container `<div>` that causes the content to appear sunken on the page (see Figure 3-50). I have used wells for blog post meta information like author, date, and categories. To create a well, simply wrap the content that you would like to appear in the well with a `<div>` containing the class of `.well`:

```
<div class="well">
  ...
</div>
```

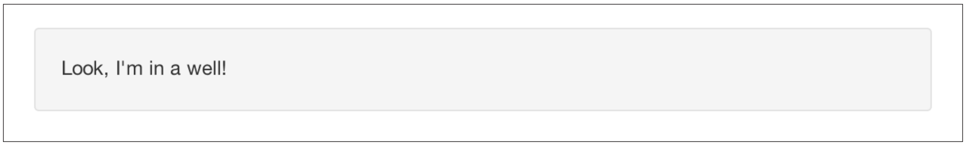
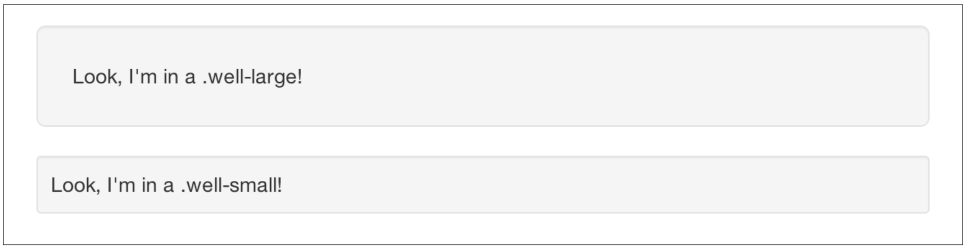

Look, I'm in a well!

Figure 3-50. Well

There are two additional classes that can be used in conjunction with `.well`: `.well-large` and `.well-small`. These affect the padding, making the well larger or smaller depending on the class (see Figure 3-51).


Look, I'm in a .well-large!

Look, I'm in a .well-small!

Figure 3-51. Well optional classes

The following code uses the well classes:

```
<div class="well well-large">

  Look, I'm in a .well-large!

</div>

<div class="well well-small">

  Look, I'm in a .well-small!

</div>
```

Helper Classes

Here are some helper classes that might come in handy.

Pull left

To float an element to the left, use the `.pull-left` class:

```
<div class="pull-left">
  ...
</div>
```

```
.pull-left {
  float: left;
}
```

Pull right

To float an element to the right, use the `.pull-right` class:

```
<div class="pull-right">
  ...
</div>
```

```
.pull-right {
  float: right;
}
```

Clearfix

To clear the float of any element, use the `.clearfix` class. When you have two elements of different sizes that are floated alongside each other, it is necessary to force the following elements in the the code below or to *clear* the preceding content. You can do this with a simple empty `<div>` with the class :of `.clearfix`:

```
<div class="clearfix"></div>
```

```
.clearfix {
  *zoom: 1;
  &:before,
  &:after {
    display: table;
    content: "";
  }
  &:after {
    clear: both;
  }
}
```